Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №2

за 1 семестр

По дисциплине: «Языки программирования»

Тема: «Наследование и виртуальные функции»

Выполнил:

Студент 2 курса

Группы ПО-4(1)

Иваненко И. Л.

Проверил:

Хацкевич М. В.

2020

Лабораторная работа №2

Наследование и виртуальные функции

Цель работы: получить практические навыки создания иерархии классов и использования статических компонентов класса.

Вариант 10

Задание:

1. Определить иерархию классов (в соответствии с вариантом).

Перечень классов: Документ, чек, квитанция, накладная.

2. Определить в классе статическую компоненту – указатель на начало связанного списка объектов и статическую функцию для просмотра списка (инициализировать вне определения класса, в глобальной области).

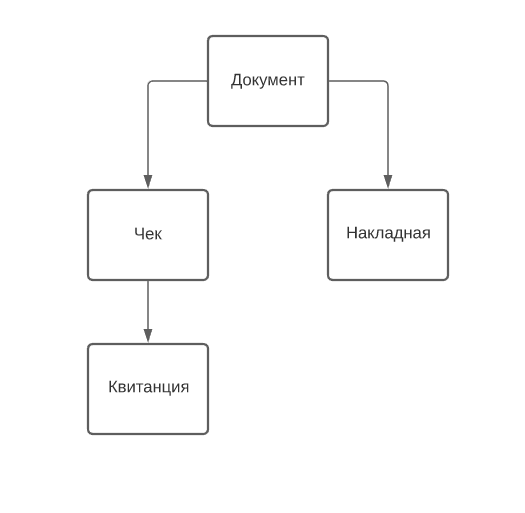
3. Реализовать классы. Определить в классах все необходимые конструкторы и деструктор.

4. Написать демонстрационную программу, в которой создаются объекты различных классов и помещаются в список, после чего список просматривается.

5. Сделать соответствующие методы не виртуальными и посмотреть, что будет.

6. Реализовать вариант, когда объект добавляется в список при создании, т.е. в конструкторе.

Иерахия классов:



Код программы:

Document.h:

#pragma once

class Document

{

protected:

float date;

int number;

public:

struct Node {

Document\* document;

Node\* next;

};

static Node\* head;

void push();

static void listOutput();

virtual void infoOutput();

void setDate(float date);

void setNumber(int number);

float getDate();

int getNumber();

Document(float date, int number);

Document();

virtual ~Document() {};

};

Document.cpp:  
#include "Document.h"

#include <iostream>

using namespace std;

Document::Node\* Document::head = NULL;

void Document::setDate(float date) {

this->date = date;

}

void Document::setNumber(int number) {

this->number = number;

}

float Document::getDate() {

return this->date;

}

int Document::getNumber() {

return this->number;

}

void Document::infoOutput() {

cout << endl << "ДОКУМЕНТ:" << endl;

cout << "Номер документа: " << this->getNumber() << endl;

cout << "Дата: " << this->getDate() << endl;

}

void Document::push() {

Node\* node = new Node;

node->document = this;

node->next = head;

head = node;

}

void Document::listOutput() {

Node\* pointer = head;

while (pointer != NULL) {

pointer->document->infoOutput();

pointer = pointer->next;

}

}

Document::Document(float date, int number) {

this->date = date;

this->number = number;

this->push();

}

Document::Document() {

this->date = 1;

this->number = 1;

this->push();

}

Check.h:

#pragma once

#include "Document.h"

class Check : public Document

{

protected:

float sum;

public:

void infoOutput() override;

void setSum(float sum);

float getSum();

Check(float date, int number, float sum);

Check();

virtual ~Check() {};

};

Check.cpp:

#include "Check.h"

#include <iostream>

using namespace std;

void Check::setSum(float sum) {

this->sum = sum;

}

float Check::getSum() {

return this->sum;

}

void Check::infoOutput() {

cout << endl << "ЧЕК:" << endl;

cout << "Номер документа: " << this->getNumber() << endl;

cout << "Дата: " << this->getDate() << endl;

cout << "Сумма: " << this->getSum() << endl;

}

Check::Check(float date, int number, float sum) {

this->date = date;

this->number = number;

this->sum = sum;

}

Check::Check() {

this->date = 1;

this->number = 1;

this->sum = 9999;

}

Waybill.h:

#pragma once

#include "Document.h"

class Waybill : public Document

{

private:

int organizationSize;

int itemSize;

int destinationSize;

protected:

char\* organization;

char\* item;

char\* destination;

float itemPrice;

public:

void infoOutput() override;

void setOrganziation(char\* organization);

void setItem(char\* item);

void setDestination(char\* destination);

void setItemPrice(float itemPrice);

char\* getOrganization();

char\* getItem();

char\* getDestination();

float getItemPrice();

Waybill(float date, char\* organization, char\* item, char\* destination, float itemPrice);

Waybill();

virtual ~Waybill();

};

Waybill.cpp:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "Waybill.h"

#include "string.h"

#include <iostream>

using namespace std;

void Waybill::setOrganziation(char\* organization) {

this->organizationSize = strlen(organization);

this->organization = new char[this->organizationSize];

strcpy(this->organization, organization);

}

void Waybill::setItem(char\* item) {

this->itemSize = strlen(item);

this->item = new char[this->itemSize];

strcpy(this->item, item);

}

void Waybill::setDestination(char\* destination) {

this->destinationSize = strlen(destination);

this->destination = new char[this->destinationSize];

strcpy(this->destination, destination);

}

void Waybill::setItemPrice(float itemPrice) {

this->itemPrice = itemPrice;

}

char\* Waybill::getOrganization() {

return this->organization;

}

char\* Waybill::getItem() {

return this->item;

}

char\* Waybill::getDestination() {

return this->destination;

}

float Waybill::getItemPrice() {

return this->itemPrice;

}

void Waybill::infoOutput() {

cout << endl << "НАКЛАДНАЯ" << endl;

cout << "Номер документа: " << this->getNumber() << endl;

cout << "Дата: " << this->getDate() << endl;

cout << "Организация: " << this->getOrganization() << endl;

cout << "Товар: " << this->getItem() << endl;

cout << "Адресат: " << this->getDestination() << endl;

cout << "Наименование товара: " << this->getItemPrice() << endl;

}

Waybill::Waybill(float date, char\* organization, char\* item, char\* destination, float itemPrice) {

this->date = date;

this->organizationSize = strlen(organization);

this->organization = new char[this->organizationSize];

strcpy(this->organization, organization);

this->itemSize = strlen(item);

this->item = new char[this->itemSize];

strcpy(this->item, item);

this->destinationSize = strlen(destination);

this->destination = new char[this->destinationSize];

strcpy(this->destination, destination);

this->itemPrice = itemPrice;

}

Waybill::Waybill() {

this->date = 10;

this->organizationSize = strlen("N/A");

this->organization = new char[this->organizationSize];

strcpy(this->organization, "N/A");

this->itemSize = strlen("N/A");

this->item = new char[this->itemSize];

strcpy(this->item, "N/A");

this->destinationSize = strlen("N/A");

this->destination = new char[this->destinationSize];

strcpy(this->destination, "N/A");

this->itemPrice = 9999;

}

Waybill::~Waybill() {

delete[] organization;

delete[] item;

delete[] destination;

}

Receipt.h:

#pragma once

#include "Check.h"

class Receipt : public Check

{

private:

int size;

protected:

char\* bank;

public:

void infoOutput() override;

void setBank(char\* bank);

char\* getBank();

Receipt(float date, int number, float sum, char\* bank);

Receipt();

~Receipt();

};

Receipt.cpp:  
#define \_CRT\_SECURE\_NO\_WARNINGS

#include "Receipt.h"

#include "string.h"

#include <iostream>

using namespace std;

void Receipt::setBank(char\* bank) {

this->size = strlen(bank);

this->bank = new char[this->size];

strcpy(this->bank, bank);

}

char\* Receipt::getBank() {

return this->bank;

}

void Receipt::infoOutput() {

cout << endl << "КВИТАНЦИЯ:" << endl;

cout << "Номер документа: " << this->getNumber() << endl;

cout << "Дата: " << this->getDate() << endl;

cout << "Сумма: " << this->getSum() << endl;

cout << "Банк: " << this->getBank() << endl;

}

Receipt::Receipt(float date, int number, float sum, char\* bank){

this->date = date;

this->number = number;

this->sum = sum;

this->size = strlen(bank);

this->bank = new char[this->size];

strcpy(this->bank, bank);

}

Receipt::Receipt() {

this->date = 1;

this->number = 1;

this->sum = 9999;

this->size = strlen("N/A");

this->bank = new char[this->size];

strcpy(this->bank, "N/A");

}

Receipt::~Receipt() {

delete[] bank;

}

Source.cpp:

#include "Document.h"

#include "Check.h"

#include "Receipt.h"

#include "Waybill.h"

#include <iostream>

using namespace std;

int main() {

setlocale(0, "");

char bank[10] = "bbbbank";

Document\* document = new Document(10, 12);

Document\* check = new Check(13, 14, 16);

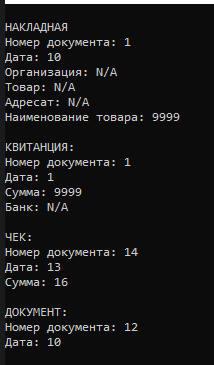
Document\* receipt = new Receipt();

Document\* waybill = new Waybill();

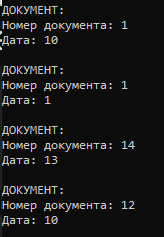
Document::listOutput();

}

Результат выполнения:



Результат выполнения(без виртуальных методов):



Вывод: В ходе выполнения данной лабораторной работы я ознакомился с наследованием в языке С++ и научился работать с абстрактными классами, методами и статическими полями классов.